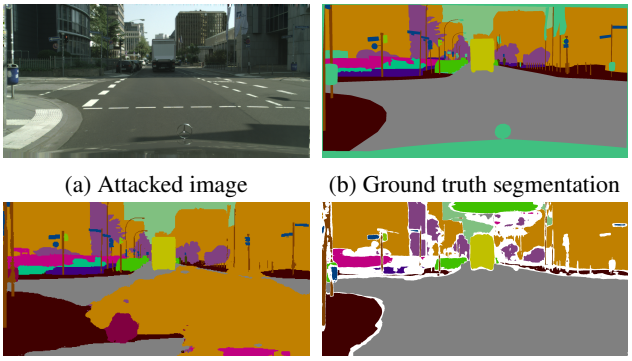
Scalable Certified Segmentation via Randomized Smoothing

Marc Fischer [1] **Maximilian Baader** [1] **Martin Vechev** [1]

Abstract

We present a new certification method for image and point cloud segmentation based on randomized smoothing. The method leverages a novel scalable algorithm for prediction and certification that correctly accounts for multiple testing, necessary for ensuring statistical guarantees. The key to our approach is reliance on established multiple-testing correction mechanisms as well as the ability to abstain from classifying single pixels or points while still robustly segmenting the overall input. Our experimental evaluation on synthetic data and challenging datasets, such as Pascal Context, Cityscapes, and ShapeNet, shows that our algorithm can achieve, for the first time, competitive accuracy and certification guarantees on real-world segmentation tasks. We provide an implementation at `https://github.com/eth-sri/segmentation-smoothing`.



(a) Attacked image (b) Ground truth segmentation

(c) Attacked segmentation (d) Certified segmentation

Figure 1. In semantic segmentation, a model segments an input (a) by classifying each pixel. While the result should match (b), a non-robust model predicts (c) as the input (a) was perturbed by additive ℓ_2 noise (PGD). Our model is certifiably robust to this perturbation (d) by abstaining from ambiguous pixels (white). The model abstains where multiple classes meet, causing ambiguity. We provide technical details and further examples in App. C.

1. Introduction

Semantic image segmentation and point cloud part segmentation are important problems in many safety critical domains including medical imaging (Perone et al., 2018) and autonomous driving (Deng et al., 2017). However, deep learning models used for segmentation are vulnerable to adversarial attacks (Xie et al., 2017; Arnab et al., 2018; Xiang et al., 2019), preventing their application to such tasks. This vulnerability is illustrated in Fig. 1, where the task is to segment the (adversarially attacked) image shown in Fig. 1a. We see that the segmentation of the adversarially attacked image is very different from the ground truth depicted in Fig. 1b, potentially causing unfavorable outcomes. While provable robustness to such adversarial perturbations is well studied for classification (Katz et al., 2017; Gehr et al., 2018; Wong & Kolter, 2018; Cohen et al., 2019), the investigation of certified segmentation just begun recently (Lorenz et al., 2021; Tran et al., 2021).

Certifiably robust segmentation is a challenging task, as the classification of each component (e.g., a pixel in an image) needs to be certified simultaneously. Many datasets and models in this domain are beyond the reach of current deterministic verification methods while probabilistic methods need to account for accumulating uncertainty over the large number of individual certifications.

In this work we propose a novel method to certify the robustness of segmentation models via randomized smoothing (Cohen et al., 2019), a probabilistic certification method able to certify ℓ_2 robustness around large images. As depicted in Fig. 1d, our method enables the certification of challenging segmentation tasks by certifying each component individually, abstaining from unstable ones that cause naive algorithms to fail. This abstention mechanism also provides strong synergies with the multiple testing correction, required for the soundness of our approach, thus enabling high certification rates.

While we focus in our evaluation on ℓ^2 robustness, our method is general and can also combined with randomized smoothing methods that certify robustness to other ℓ^p-bounded attacks or parametrized transformations like rotations (Fischer et al., 2020).

[1]Department of Computer Science, ETH Zurich, Switzerland. Correspondence to: Marc Fischer <marc.fischer@inf.ethz.ch>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

Main Contributions Our key contributions are:

- We investigate the obstacles of scaling randomized smoothing from classification to segmentation, identifying two key challenges: the influence of single bad components and the multiple testing trade-offs (§4)
- We introduce a scalable algorithm that addresses these issues, allowing, for the first time, to certify large scale segmentation models (§5).
- We show that this algorithm can be applied to different generalizations of randomized smoothing, enabling defenses against different attacker models (§5.2).
- We provide an extensive evaluation on semantic image segmentation and point cloud part segmentation, achieving up to 88% and 55% certified pixel accuracy on Cityscapes and Pascal context respectively, while obtaining mIoU of 0.6 and 0.2 (§6).

2. Related Work

In the following, we survey the most relevant related work.

Adversarial Attacks Biggio et al. (2013); Szegedy et al. (2014) discovered adversarial examples, which are inputs perturbed in a way that preserves their semantics but fools deep networks. To improve performance on these inputs, training can be extended to including adversarial examples, called adversarial training (Kurakin et al., 2017; Madry et al., 2018). Most important to this work are attacks on semantic segmentation: Xie et al. (2017) introduced a targeted gradient based unconstrained attack, by maximizing the summed individual targeted losses for every pixel. Arnab et al. (2018) applied FGSM based attacks (Goodfellow et al., 2015) to the semantic segmentation setting. Similarly, the vulnerability of point cloud classifiers was exposed in Xiang et al. (2019); Liu et al. (2019); Sun et al. (2020).

Certified robustness and defenses However, adversarially trained neural networks do not come with robustness guarantees. To address this, different certification methods have been proposed recently using various methods, relying upon SMT solvers (Katz et al., 2017; Ehlers, 2017), semidefinite programming (Raghunathan et al., 2018a) and linear relaxations (Gehr et al., 2018; Zhang et al., 2018; Wang et al., 2018; Weng et al., 2018; Wong & Kolter, 2018; Singh et al., 2019b;a). Specifically, linear relaxations have been used beyond the classical ℓ_p noise setting to certify against geometric transformations (Singh et al., 2019b; Balunovic et al., 2019; Mohapatra et al., 2020b) and vector field attacks (Ruoss et al., 2021).

To further improve certification rates, methods that train networks to be certifiable have been proposed (Raghunathan et al., 2018b; Mirman et al., 2018; Gowal et al., 2018; Balunovic & Vechev, 2020).

Notable to our setting are Lorenz et al. (2021); Tran et al. (2021), who extend deterministic certification to point cloud segmentation and semantic segmentation respectively. However, due to the limitations of deterministic certification these models are small in scale. Further, Bielik & Vechev (2020); Sheikholeslami et al. (2021) improved robust classifiers with the ability to abstain from classification.

Randomized Smoothing Despite all this, deterministic certification performance on complicated datasets remained unsatisfactory. Recently, based on Li et al. (2018) and Lécuyer et al. (2019), Cohen et al. (2019) presented randomized smoothing, which was the first certification method to successfully certify ℓ_2 robustness of large neural networks on large images. Salman et al. (2019) improved the results, by combining the smoothing training procedure with adversarial training. Yang et al. (2020) derive conditions for optimal smoothing distributions for ℓ_1, ℓ_2 and ℓ_∞ adversaries, if only label information is available. Mohapatra et al. (2020a) incorporated gradient information to improve certification radii. Zhai et al. (2020); Jeong & Shin (2020) improved the training procedure for base models by introducing regularized losses.

Randomized smoothing has been extended in various ways. Bojchevski et al. (2020) proposed a certification scheme suitable for discrete data and applied it successfully to certify graph neural networks. (Levine & Feizi, 2020b) certified robustness against Wasserstein adversarial examples. Fischer et al. (2020) and Li et al. (2021) used randomized smoothing to certify robustness against geometric perturbations. Salman et al. (2020) showed that using a denoiser, of the shelf classifiers can be turned into certifiable classifiers without retraining. Levine & Feizi (2020a) and Lin et al. (2021) presented methods to certify robustness against adversarially placed patches.

Most closely related to this work are Chiang et al. (2020), which introduces median smoothing and applies it to certify object detectors, and Schuchardt et al. (2021), which also extends randomized smoothing to collective robustness certificates over multiple components. They specifically exploit the locality of the classifier to the data, making their defense particularly suitable for graphs where an attacker can only modify certain subsets. While their approach can in principle be applied to semantic segmentation, modern approaches commonly rely on global information.

3. Randomized Smoothing for Classification

In this section we will briefly review the necessary background and notation on randomized smoothing, before ex-

Algorithm 1 adapted from (Cohen et al., 2019)

 # evaluate $\bar{f}$ at $\boldsymbol{x}$
function PREDICT($f, \sigma, \boldsymbol{x}, n, \alpha$)
  $\texttt{cnts} \leftarrow$ SAMPLE($f, \boldsymbol{x}, n, \sigma$)
  $\hat{c}_A, \hat{c}_B \leftarrow$ top two indices in $\texttt{cnts}$
  $n_A, n_B \leftarrow \texttt{cnts}[\hat{c}_A], \texttt{cnts}[\hat{c}_B]$
  if BINPVALUE($n_A, n_A + n_B, =, 0.5$) $\leq \alpha$ **return** $\hat{c}_A$
  else return $\oslash$

 # certify the robustness of $\bar{f}$ around $\boldsymbol{x}$
function CERTIFY($f, \sigma, \boldsymbol{x}, n_0, n, \alpha$)
  $\texttt{cnts}^0 \leftarrow$ SAMPLE($f, \boldsymbol{x}, n_0, \sigma$)
  $\hat{c}_A \leftarrow$ top index in $\texttt{cnts}^0$
  $\texttt{cnts} \leftarrow$ SAMPLE($f, \boldsymbol{x}, n, \sigma$)
  $\underline{p_A} \leftarrow$ LOWERCONFBND($\texttt{cnts}[\hat{c}_A], n, 1-\alpha$)
  if $\underline{p_A} > \frac{1}{2}$ **return** prediction $\hat{c}_A$ and radius $\sigma\,\Phi^{-1}(\underline{p_A})$
  else return $\oslash$

tending it in §4 and §5. Randomized smoothing (Cohen et al., 2019) constructs a robust (smoothed) classifier $\bar{f}$ from a (base) classifier f. The classifier $\bar{f}$ is then provably robust to ℓ_2-perturbations up to a certain radius.

Concretely, for a classifier $f\colon \mathbb{R}^m \mapsto \mathcal{Y}$ and random variable $\epsilon \sim \mathcal{N}(0, \sigma^2\mathbb{1})$: we define a smoothed classifier $\bar{f}$ as

$$\bar{f}(\boldsymbol{x}) := \arg\max_c \mathbb{P}_{\epsilon\sim\mathcal{N}(0,\sigma^2\mathbb{1})}(f(\boldsymbol{x}+\epsilon) = c). \quad (1)$$

This classifier $\bar{f}$ is then robust to adversarial perturbations:

Theorem 3.1 (From (Cohen et al., 2019))**.** *Suppose $c_A \in \mathcal{Y}$, $\underline{p_A}, \overline{p_B} \in [0,1]$. If*

$$\mathbb{P}_\epsilon(f(\boldsymbol{x}+\epsilon) = c_A) \geq \underline{p_A} \geq \overline{p_B} \geq \max_{c\neq c_A} \mathbb{P}_\epsilon(f(\boldsymbol{x}+\epsilon) = c),$$

then $\bar{f}(\boldsymbol{x}+\delta) = c_A$ for all δ satisfying $\|\delta\|_2 \leq R$ with $R := \frac{\sigma}{2}(\Phi^{-1}(\underline{p_A}) - \Phi^{-1}(\overline{p_B}))$.

In order to evaluate the model $\bar{f}$ and calculate its robustness radius R, we need to be able to compute $\underline{p_A}$ for the input $\boldsymbol{x}$. However, since the computation of the probability in Eq. (1) is not tractable (for most choices of f) $\bar{f}$ can not be evaluated exactly. To still query it, Cohen et al. (2019) suggest PREDICT and CERTIFY (Algorithm 1) which use Monte-Carlo sampling to approximate $\bar{f}$. Both algorithms utilize the procedure SAMPLE, which samples n random realizations of $\epsilon \sim \mathcal{N}(0, \sigma)$ and computes $f(\boldsymbol{x}+\epsilon)$, which is returned as a vector of counts for each class in $\mathcal{Y}$. These samples are then used to estimate the class c_A and radius R with confidence $1-\alpha$, where $\alpha \in [0,1]$. PREDICT utilizes a two-sided binomial p-value test to determine $\bar{f}(\boldsymbol{x})$. With probability at most α it will abstain, denoted as $\oslash$ (a predefined value), else it will produce $\bar{f}(\boldsymbol{x})$. CERTIFY uses the Clopper-Pearson confidence interval (Clopper & Pearson, 1934) to soundly estimate $\underline{p_A}$ and then invoke Theorem 3.1 with $\overline{p_B} = 1 - \underline{p_A}$ to obtain R. If CERTIFY returns a class other than $\oslash$ and a radius R, then with probability $1-\alpha$ the guarantee in Theorem 3.1 holds for this R. The certification radius R increases if (i) the value of $\underline{p_A}$ increases, which increases if the classifier f is robust to noise, (ii) the number of samples n used to estimate it increases, or (iii) the α increases which means that the confidence decreases.

4. Randomized Smoothing for Segmentation

To show how randomized smoothing can be applied in the segmentation setting we first discuss the mathematical problem formulation and two direct adaptations of randomized smoothing to the problem. By outlining how these fail in practice, we determine the two key challenges preventing their success. Then in, §5, we address these challenges.

Segmentation Given an input $\boldsymbol{x} = \{\boldsymbol{x}_i\}_{i=0}^N$ of N components $\boldsymbol{x}_i \in \mathcal{X}$ (e.g., points or pixels) and a set of possible classes $\mathcal{Y}$, segmentation can be seen as a function $f\colon \mathcal{X}^N \to \mathcal{Y}^N$. That is, to each $\boldsymbol{x}_i$ we assign a $f_i(\boldsymbol{x}) = y_i \in \mathcal{Y}$, where f_i denotes the i-th component of the output of f invoked on input $\boldsymbol{x}$. Here we assume $\mathcal{X} := \mathbb{R}^m$. Unless specified, we will use $m = 3$ as this allows for RGB color pixels as well as 3d point clouds.

Direct Approaches To apply randomized smoothing, as introduced in §3, to segmentation, we can reduce it to one or multiple classification problems.

We can recast segmentation $f\colon \mathcal{X}^N \to \mathcal{Y}^N$ as a classification problem by considering the cartesian product of the co-domain $\mathcal{V} := \bigtimes_{i=1}^N \mathcal{Y}$ and a new function $f'\colon \mathcal{X}^N \to \mathcal{V}$ that performs classification. Thus we can apply CERTIFY (Algorithm 1) to the base classifier f'. This provides a mathematically sound mapping of segmentation to classification. However, a change in the classification of a single component $\boldsymbol{x}_i$ will change the overall class in $\mathcal{V}$ making it hard to find a majority class $\hat{c}_A$ with high $\underline{p_A}$ in practice. We refer to this method as JOINTCLASS, short for "joint classification".

Alternatively, rather than considering all components at the same time, we can also classify each component individually. To this end, we let $f_i(\boldsymbol{x})$ denote the i-th component of $f(\boldsymbol{x})$ and apply CERTIFY, in Algorithm 1, N times to evaluate $\tilde{f}_i(\boldsymbol{x})$ to obtain classes $\hat{c}_{A,1}, \ldots, \hat{c}_{A,N}$ and radii $R_1, \ldots, R_N$. Then the overall radius is given as $R = \min_i R_i$ and a single abstention will cause an overall abstention. To reduce evaluation cost we can reuse the same input samples for all components of the output vector, that is sample $f(\boldsymbol{x})$ rather than individual $f_i(\boldsymbol{x})$. We refer to this method as INDIVCLASS. Further, the result of each call to CERTIFY only holds with probability $1-\alpha$. Thus, using the union bound, the overall

correctness is limited by $1 - \mathbb{P}(\bigvee_i i\text{-th test incorrect}) \leq 1 - \max(\sum_i \mathbb{P}(i\text{-th test incorrect}), 1) = 1 - \min(N\alpha, 1)$ (by the union bound), which for large N quickly becomes problematic. This can be compensated by carrying out calls to CERTIFY with $\alpha' = \frac{\alpha}{N}$, which becomes prohibitively expensive as n needs to be increased for the procedure to not abstain.

Key Challenges These direct applications of randomized smoothing to segmentation suffer from multiple problems that can be reduced to two challenges:

- *Bad Components*: Both algorithms can be forced to abstain or report a small radius by a single bad component $\boldsymbol{x}_i$ for which the base classifier is unstable.
- *Multiple Testing Trade-off*: Any algorithm that, like INDIVCLASS, reduces the certification segmentation to multiple stochastic tests (such as CERTIFY) suffers from the multiple testing problem. As outlined before, if each partial result only holds with probability α then the overall probability decays, using the union bound, linearly in the number of tests. Thus, to remain sound one is forced to choose between scalability or low confidence.

5. Scalable Certified Segmentation

We now introduce our algorithm for certified segmentation by addressing these challenges. In particular, we will sidestep the bad component issue and allow for a more favorable trade-off in the multiple-testing setting.

To limit the impact of bad components on the overall result, we introduce a threshold $\tau \in [\frac{1}{2}, 1)$ and define a model $\bar{f}^\tau : \mathcal{X}^N \to \hat{\mathcal{Y}}^N$, with $\hat{\mathcal{Y}} = \mathcal{Y} \cup \{\oslash\}$, that abstains if the probability of the top class for component $\boldsymbol{x}_i$ is below τ:

$$\bar{f}_i^\tau(\boldsymbol{x}) = \begin{cases} c_{A,i} & \text{if } \mathbb{P}_{\epsilon\sim\mathcal{N}(0,\sigma)}(f_i(\boldsymbol{x}+\epsilon)) > \tau \\ \oslash & \text{else} \end{cases},$$

where $c_{A,i} = \arg\max_{c\in\mathcal{Y}} \mathbb{P}_{\epsilon\sim\mathcal{N}(0,\sigma)}(f_i(\boldsymbol{x}+\epsilon) = c)$.

This means that on components with fluctuating classes, the model $\bar{f}^\tau$ does not need to commit to a class. For the model $\bar{f}^\tau$, we obtain a safety guarantee similar to the original theorem by (Cohen et al., 2019):

Theorem 5.1. *Let $\mathcal{I}_{\boldsymbol{x}} = \{i \mid \bar{f}_i^\tau(\boldsymbol{x}) \neq \oslash, i \in 1, \ldots, N\}$ denote the set of non-abstain indices for $\bar{f}^\tau(\boldsymbol{x})$. Then,*

$$\bar{f}_i^\tau(\boldsymbol{x}+\delta) = \bar{f}_i^\tau(\boldsymbol{x}), \quad \forall i \in \mathcal{I}_{\boldsymbol{x}}$$

for $\delta \in \mathbb{R}^{N\times m}$ with $\|\delta\|_2 \leq R := \sigma\Phi^{-1}(\tau)$.

Proof. We consider $\bar{f}_1^\tau, \ldots, \bar{f}_N^\tau$ independently. With $p_{A,i} := \mathbb{P}_{\epsilon\sim\mathcal{N}(0,\sigma)}(f_i(\boldsymbol{x}+\epsilon) = c_{A,i}) > \tau$, we invoke Theorem 3.1 with $\underline{p_A} = \tau$ for $\bar{f}^\tau$ and obtain robustness radius $R := \sigma\Phi^{-1}(\tau)$ for $\bar{f}_i^\tau(\boldsymbol{x})$. This holds for all i where the class probability $p_{A,i} > \tau$, denoted by the set $\mathcal{I}_{\boldsymbol{x}}$. □

Algorithm 2 algorithm for certification and prediction

 # evaluate $\bar{f}^\tau$ at $\boldsymbol{x}$
function SEGCERTIFY(f, σ, $\boldsymbol{x}$, n, n_0, τ, α)
  $\texttt{cnts}_1^0, \ldots, \texttt{cnts}_N^0 \leftarrow$ SAMPLE(f, $\boldsymbol{x}$, n_0, σ)
  $\texttt{cnts}_1, \ldots, \texttt{cnts}_N \leftarrow$ SAMPLE(f, $\boldsymbol{x}$, n, σ)
  for $i \leftarrow \{1, \ldots, N\}$:
    $\hat{c}_i \leftarrow$ top index in $\texttt{cnts}_i^0$
    $n_i \leftarrow \texttt{cnts}_i[\hat{c}_i]$
    $pv_i \leftarrow$BINPVALUE(n_i, n, $\leq$, τ)
  $r_1, \ldots, r_N \leftarrow$ FWERCONTROL(α, $pv_1, \ldots, pv_N$)
  for $i \leftarrow \{1, \ldots, N\}$:
    if $\neg r_i$: $\hat{c}_i \leftarrow \oslash$
  $R \leftarrow \sigma\Phi^{-1}(\tau)$
  return $\hat{c}_1, \ldots, \hat{c}_N$, R

Certification Similarly to Cohen et al. (2019), we cannot invoke our theoretically constructed model $\bar{f}^\tau$ and must approximate it. The simplest way to do so would be to invoke CERTIFY for each component and replace the check $\underline{p_A} > \frac{1}{2}$ by $\underline{p_A} > \tau$. However, while this accounts for the bad component issue, it still suffers from the outlined multiple testing problem. To address this issue, we now introduce the SEGCERTIFY procedure in Algorithm 2.

We will now briefly outline the steps in Algorithm 2 before describing them in further detail and arguing about its correctness and properties. Algorithm 2 relies on the same primitives as Algorithm 1, as well as the FWERCONTROL function, which performs multiple-testing correction, which we will formally introduce shortly. As before, SAMPLE denotes the evaluation of samples $f(\boldsymbol{x}+\epsilon)$ and $\texttt{cnts}_i$ denotes the class frequencies observed for the i-th component. Similar to CERTIFY, we use two sets of samples $\texttt{cnts}$ and $\texttt{cnts}^0$ to avoid a model selection bias (and thus invalidate our statistical test): We use n_0 samples, denoted $\texttt{cnts}_i^0$ to guess the majority class $c_{A,i}$ for the i-th component, denoted $\hat{c}_i$ and then determine its appearance count n_i out of n trails from $\texttt{cnts}_i$. With these counts, we then perform a one-sided binomial test, discussed shortly, to obtain its p-value pv_i. Given these p-values we can employ FWERCONTROL, which determines which tests we can reject in order to obtain overall confidence $1-\alpha$. The rejection of the i-th test is denoted by the boolean variable r_i. If we reject the i-th test ($r_i = 1$), we assume its alternate hypothesis $p_{A,i} > \tau$ and return the class $c_{A,i}$, else we return $\oslash$.

To establish the correctness of the approach and improve on the multiple testing trade-off, we will now briefly review statistical (multiple) hypotheses testing.

Hypothesis Testing Here we consider a fixed but arbitrary single component i: If we guessed the majority class $c_{A,i}$ for the i-th component to be $\hat{c}_i$, we assume that $f_i(\boldsymbol{x}+\epsilon)$ returns $\hat{c}_i$ with probability $p_{\hat{c}_i,i}$ (denoted p in the following). We now want to infer wether $p > \tau$ or not, to determine whether $\bar{f}^\tau$ will output $\hat{c}_i$ or $\oslash$. We phrase this as a statistical test with null hypothesis $(H_{0,i})$ and alternative $(H_{A,i})$:

$$(H_{0,i}):\ p \leq \tau \qquad\qquad (H_{A,i}):\ p > \tau$$

A statistical test lets us assume a null hypothesis $(H_{0,i})$ and check how plausible it is given our observational data. We can calculate the p-value pv, which denotes the probability of the observed data or an even more extreme event under the null hypothesis. Thus, in our case rejecting $(H_{0,i})$ means returning $\hat{c}_i$ while accepting it means returning $\oslash$.

Rejecting the null hypothesis when it is actually true (in our case returning a class if we should abstain) is called a type I error. The opposite, not rejecting the null hypothesis when it is false is called a type II error. In our setting type II errors mean additional abstentions on top of those $\bar{f}^\tau$ makes by design. Making a sound statement about robustness means controlling the type I error while reducing type II errors means fewer abstentions due to the testing procedure.

Commonly, a test is rejected if its p-value is below some threshold α, as in the penultimate line of PREDICT (Algorithm 1). This bounds the probability of type I error at probability (or "level") α.

Multiple Hypothesis Testing Multiple tests performed at once require additional care. Usually, if we perform N tests, we want to bound the probability of any type I error occurring. This probability is called the *family wise error rate* (FWER). The goal of FWER control is, given a set of N tests and α, to reject tests such that the FWER is limited at α. In Algorithm 2 this is denoted as FWERCONTROL, which is a procedure that takes the value α and p-values $pv_1, \ldots, pv_N$ and decides on rejections $r_1, \ldots, r_N$.

The simplest procedure for this is the Bonferroni method (Bonferroni, 1936), which rejects individual tests with p-value $pv_i \leq \frac{\alpha}{N}$. It is based on the same consideration we applied for the failure probability of INDIVCLASS in §4. While this controls the FWER at level α it also increases the type II error, and thus would reduce the number of correctly classified components. To maintain the same number of correctly classified components we would need to increase n or decrease α.

A better alternative to the Bonferroni method is the Holm correction (or Holm-Bonferroni method) (Holm, 1979) which orders the tests by acceding p-value and steps through them at levels $\frac{\alpha}{N}, \ldots, \frac{\alpha}{1}$ (rejecting null if the associated p value is smaller then the level) until the first level where no additional test can be rejected. This method also controls the FWER at level α but allows for a lower rate of type II errors.

Further improving upon these methods, usually requires additional information. This can either be specialization to a certain kind of test (Tukey, 1949), additional knowledge (e.g., no negative dependencies between tests, for procedures such as (Šidák, 1967)) or an estimate of the dependence structure computed via Bootsrap or permutation methods (Westfall & Young, 1993).

While our formulation of SEGCERTIFY admits all of these corrections, Bootstrap or permutaion procedures are generally infeasible for large N. Thus, in the rest of the paper we consider both Holm and Bonferroni correction and compare them in §6.1.

We note that while methods (Westfall, 1985) exist for assessing that N Clopper-Pearson confidence intervals hold jointly at level α, thus allowing better correction for INDIVCLASS, these methods do not scale to realistic segmentation problems as they require re-sampling operations that are expensive on large problems.

5.1. Properties of SEGCERTIFY

SEGCERTIFY is a conservative algorithm, as it will rather abstain from classifying a component than returning a wrong or non-robust result. We formalize this as well as a statement about the correctness of SEGCERTIFY in Proposition 1.

Proposition 1. *Let $\hat{c}_1, \ldots, \hat{c}_N$ be the output of* SEGCERTIFY *for input $\boldsymbol{x}$ and $\hat{\mathcal{I}}_{\boldsymbol{x}} := \{i \mid \hat{c}_i \neq \oslash\}$. Then with probability at least $1-\alpha$ over the randomness in* SEGCERTIFY *$\hat{\mathcal{I}}_{\boldsymbol{x}} \subseteq \mathcal{I}_{\boldsymbol{x}}$, where $\mathcal{I}_{\boldsymbol{x}}$ denotes the previously defined non-abstain indices of $\bar{f}^\tau(\boldsymbol{x})$. Therefore $\hat{\boldsymbol{c}}_i = \bar{f}_i^\tau(\boldsymbol{x}) = \bar{f}_i^\tau(\boldsymbol{x}+\delta)$ for $i \in \hat{\mathcal{I}}_{\boldsymbol{x}}$ and $\delta \in \mathbb{R}^{N\times m}$ with $\|\delta\|_2 \leq R$.*

Proof. Suppose that $i \in \hat{\mathcal{I}}_{\boldsymbol{x}} \setminus \mathcal{I}_{\boldsymbol{x}}$. This i then represents a type I error. However, the overall probability of any such type I error is controlled at level α by the FWERCONTROL step. Thus with probability at least $1-\alpha$, $\hat{\mathcal{I}}_{\boldsymbol{x}} \subseteq \mathcal{I}_{\boldsymbol{x}}$. The rest follows from Theorem 5.1. □

We note that there are fundamentally two reasons for SEGCERTIFY to abstain from the classification of a component: either because the true class probability $p_{A,i}$ is less than or equal to τ in which case $\bar{f}_i^\tau$ abstains by definition or because of a type II error in SEGCERTIFY. This type II error can occur either if we guess the wrong $\hat{c}_i$ from our n_0 samples or because we could not gather sufficient evidence to reject $(H_{0,i})$ under α-FWER. Proposition 1 only makes a statement about type I errors, but not type II errors, e.g., $\hat{\mathcal{I}}$ could always be the empty set. In general, the control of type II errors (called "power") of a testing procedure can

be hard to quantify without a precise understanding of the underlying procedure (in our case the model f). However, in §6.1 we will investigate this experimentally.

In contrast to Cohen et al. (2019) we do not provide separate procedures for prediction and certification, as for a fixed τ both only need to determine $p_{A,i} > \tau$ for all i with high probability (w.h.p.). For faster prediction, the algorithm can be invoked with larger α and smaller n.

For $N = 1$, the algorithm exhibits the same time complexity as those proposed by Cohen et al. (2019). For larger N, time complexity stays the same (plus the time for sorting the p-values in Holm procedure), however due to type II errors the number of abstentions increases. Counteracting this leads to the previously discussed multiple-testing trade-off.

Summary Having presented SEGCERTIFY we now summarize how it addresses the two key challenges. First, the bad component issue is offset by the introduction of a threshold parameter τ. This allows to exclude single components, which would not be provable, from the overall certificate. Second, due to this threshold we now aim to show a fixed lower bound $\underline{p_A} = \tau$, where we can use a binomial p-value test rather than the Clopper-Pearson interval for a fixed confidence. The key difference here is that the binomial test produces a p-value that is small unless the observed rate $\frac{n_i}{n}$ is close to τ, while the Clopper-Pearson interval aims to show the maximal $\underline{p_A}$ for a fixed confidence. This former setting generally lends itself better to the multiple testing setting. In particular for correction algorithms such as Holm this lets small p-values offset larger ones.

5.2. Generality

While presented in the ℓ_2-robustness setting as in Cohen et al. (2019), SEGCERTIFY is not tied to this and is compatible with many extensions and variations of Theorem 3.1 in a plug-and-play manner. For example, Li et al. (2018); Lécuyer et al. (2019); Yang et al. (2020) showed robustness to ℓ_1 (and other ℓ_p) perturbations. Fischer et al. (2020) introduced an extension that allows computing a robustness radius over the parameter of a parametric transformation. In both of these, it suffices for practical purposes to update SAMPLE to produce perturbations from a suitable distribution (e.g., rotation or Laplace noise) as well as update the calculation of the radius R to reflect the different underlying theorems. Similarly, in $\bar{f}^\tau$ and Theorem 5.1, it suffices to update the class probability according to the sample distribution and radius prescribed by the relevant theorem.

Extensions to k-FWER When we discussed FWER control so far, we bounded the probability of making any type I error by α. However, in the segmentation setting we have many individual components, and depending on our objective, we may allow a small budget of few errors. In this setting, we can employ k-FWER control (Lehmann & Romano, 2012). It controls the probability of observing k or more type I errors at probability α. Thus with probability $1 - \alpha$ we have observed at most $k - 1$ type I errors (false non-abstentions). Lehmann & Romano (2012) introduces a procedure similar to Holm method that allows for k-FWER that is optimal (without the use of further information on test dependence) and can be simply plugged into SEGCERTIFY. We provide further explanation and evaluation in App. B.2.

6. Experimental Evaluation

We evaluate our approach in three different settings: (i) investigating the two key challenges on toy data in §6.1, (ii) semantic image segmentation in §6.2, and (iii) point clouds part segmentation in §6.3. All timings are for a single Nvidia GeForce RTX 2080 Ti. For further details, see App. A.

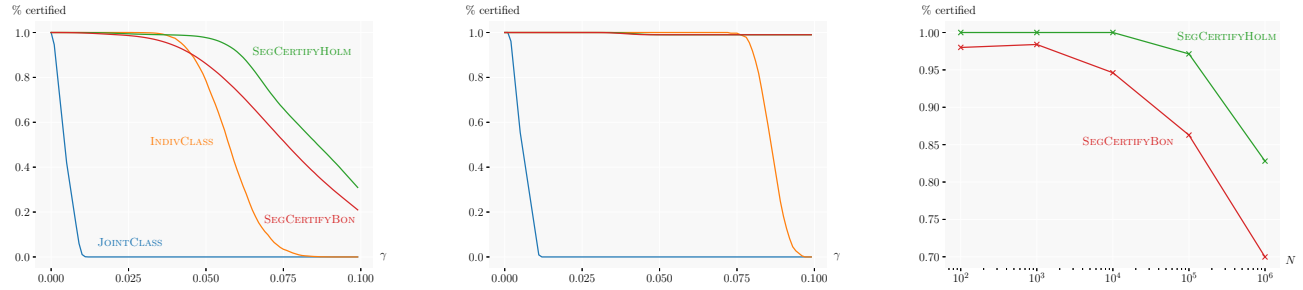
6.1. Toy Data

We now investigate how well SEGCERTIFY and the two naive algorithms handle the identified challenges.

Here we consider a simple setting, where our input is of length N and each component can be from two possible classes. Further, our base model is an oracle (that does not need to look at the data) and is correct for each of $N - k$ component with probability $1 - \gamma$ for some $\gamma \in [0, 1]$ and $k \in 0, \dots, N$. On the remaining k components it is correct with probability $1 - 5\gamma$ (clamped to $[0, 1]$).

We evaluate JOINTCLASS, INDIVCLASS with Bonferroni correction and SEGCERTIFY with Holm, denoted SEGCERTIFYHOLM, as well as Bonferroni correction, denoted SEGCERTIFYBON. Note that SEGCERTIFYBON in contrast to INDIVCLASS employs thresholding. We investigate the performance of all methods under varying levels of noise by observing the rate of certified components. As the naive algorithms either provide a certificate for all pixels or none, we have repeated the experiment 600 times, showing a smoothed average in Fig. 2a (the unsmoothed version can be found in App. A). In this setting, we assume $N = 100$ components, $k = 1$ and γ varies between 0 and 0.1 along the x-axis. All algorithms use $n_0 = n = 100$ and $\alpha = 0.001$. SEGCERTIFY uses $\tau = 0.75$. This means that for $\gamma \leq 0.05$, all abstentions are due to the testing procedure (type II errors) and not prescribed by τ in the definition of $\bar{f}^\tau$. For $0.05 < \gamma \leq 0.1$, there is one prescribed abstention as $k = 1$. Thus the results provide an empirical assessment of the statistical power of the algorithm.

As outlined §4, JOINTCLASS and INDIVCLASS are very susceptible to a single bad component. While JOINTCLASS almost fails immediately for $\gamma > 0$, INDIVCLASS works until γ becomes large enough so that the estimated confi-



(a) On $N = 100$ components, with a classifier that has error rate 5γ on one component and γ on all others.

(b) Same setting as Fig. 2a, but the algorithms use more samples n_0, n allowing for less abstentions.

(c) SegCertify with different testing corrections for various numbers of components N with error rate $\gamma = 0.05$.

Figure 2. We empirically investigate the power (ability to avoid type II errors – false abstention) of multiple algorithms on synthetic data. The y-axis shows the rate of certified (rather than abstained) components. An optimal algorithm would achieve 1.0 or 0.99 in all plots.

dence interval of $1 - 5\gamma$ cannot be guaranteed to be larger than 0.5. Both variants of SegCertify also deteriorate with increasing noise. The main culprit for this is guessing a wrong class in the n_0 samples. We showcase this in Fig. 2b, where we use the same setting as in Fig. 2a but with $n_0 = n = 1000$. Here, both SegCertify variants have the expected performance (close to 1) even with increasing noise. For IndivClass, these additional samples also delay complete failure, but do not prevent it. The main takeaway from Figs. 2a and 2b is that SegCertify overcomes the bad component issue and the number of samples n, n_0 is a key driver for its statistical power.

Lastly, in Fig. 2c we compare SegCertifyBon and SegCertifyHolm. Here, $\gamma = 0.05$, $k = 0$, $n = n_0 = 1000$, $\alpha = 0.1$ and N increases (exponentially) along the x-axis. We see both versions deteriorate in performance as N grows. As before, this is to be expected as out of N components some of the initial guesses based on n_0 samples may be wrong or n may be too small to correctly reject the test. However, we note that the certification gap between SegCertifyBon and SegCertifyHolm grows as N increases due to their difference in statistical power.

We conclude that SegCertify addresses both challenges, solving the bad component issue and achieving better trade-offs in multiple testing. Particularly in the limit the benefit of Holm correction over Bonferrroni becomes apparent.

6.2. Semantic Image Segmentation

Next, we evaluate SegCertify on the task of semantic image segmentation considering two datasets, Cityscapes, and Pascal Context. The Cityscapes dataset (Cordts et al., 2016) contains high-resolution (1024×2048 pixel) images of traffic scenes annotated in 30 classes, 19 of which are used for evaluation. An example of this can be seen in Fig. 1 (the hood of the car is in an unused class). The Pascal Context dataset (Mottaghi et al., 2014) contains images with 59 foreground and 1 background classes, which before segmentation are resized to 480×480. There are two common evaluation schemes, either using all 60 classes or just the 59 foreground classes. Here we use the later setting.

As a base model in this settings, we use a HrNetV2 (Sun et al., 2019; Wang et al., 2019). Like many segmentation models, it can be invoked on different scales. For example, to achieve faster inference we scale an image down half its length and width, invoke the segmentation model, and scale up the result. Or, to achieve a more robust or more accurate segmentation we can invoke segmentation on multiple scales, scale the the output probabilities to the original input size, average them, and then predict the class for each pixel. Here we investigate how different scales allow different trade-offs for provably robust segmentation via SegCertify. We note that at the end, we always perform certification on the 1024×2048-scale result. We trained our base models with Gaussian Noise ($\sigma = 0.25$), as in Cohen et al. (2019). Details about the training procedure and further results can be found in App. A.1 and B.3.

Evaluation results for 100 images on both datasets are given in Table 1. We observe that the certified model has accuracy close to that of the base model, even outperforming it sometimes, even though it is abstaining for a number of pixels. This means that abstentions generally happens in wrongly predicted areas or on ignored classes. Similar to Cohen et al. (2019), we observe best performance on the noise level we trained the model on ($\sigma = 0.25$) and degrading performance for higher values. Interestingly, by comparing the results for $\sigma = 0.5$ at scale 1.0 and smaller scales, we observe that the resizing for scales < 1.0 acts as a natural denoising step, allowing better performance with more noise. Further, in Fig. 3 we show how the certified accuracy degrates with different values of τ. We use scale 0.25 and $n = 300$ and the same parameters as in Table 1 otherwise. Up to $\tau = 0.92$ we observe very gradual drop-off with increasing τ. Then

Table 1. Segmentation results for 100 images. acc. shows the mean per-pixel accuracy, mIoU the mean intersection over union, %⊘ abstentions and t runtime in seconds. All SEGCERTIFY ($n_0 = 10, \alpha = 0.001$) results are certifiably robust at radius R w.h.p. multiscale uses $0.5, 0.75, 1.0, 1.25, 1.5, 1.75$ as well as their flipped variants for Cityscapes and additionally 2.0 for Pascal.

				Cityscapes				Pascal Context			
scale		σ	R	acc.	mIoU	%⊘	t	acc.	mIoU	%⊘	t
0.25	non-robust model	-	-	0.93	0.60	0.00	0.38	0.59	0.24	0.00	0.12
	base model	-	-	0.87	0.42	0.00	0.37	0.33	0.08	0.00	0.13
	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	0.84	0.43	0.07	70.00	0.33	0.08	0.13	14.16
		0.33	0.22	0.84	0.44	0.09	70.21	0.34	0.09	0.17	14.20
		0.50	0.34	0.82	0.43	0.13	71.45	0.23	0.05	0.27	14.23
	SEGCERTIFY $n = 500, \tau = 0.95$	0.25	0.41	0.83	0.42	0.11	229.37	0.29	0.01	0.30	33.64
		0.33	0.52	0.83	0.42	0.12	230.69	0.26	0.01	0.39	33.79
		0.50	0.82	0.77	0.38	0.20	230.09	0.10	0.00	0.61	33.44
0.5	non-robust model	-	-	0.96	0.76	0.00	0.39	0.74	0.38	0.00	0.16
	base model	-	-	0.89	0.51	0.00	0.39	0.47	0.13	0.00	0.14
	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	0.88	0.54	0.06	75.59	0.48	0.16	0.09	16.29
		0.33	0.22	0.87	0.54	0.08	75.99	0.50	0.17	0.11	16.08
		0.50	0.34	0.86	0.54	0.10	75.72	0.36	0.10	0.21	16.14
1.0	non-robust model	-	-	0.97	0.81	0.00	0.52	0.77	0.42	0.00	0.18
	base model	-	-	0.91	0.57	0.00	0.52	0.53	0.18	0.00	0.18
	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	0.88	0.59	0.11	92.75	0.55	0.22	0.22	18.53
		0.33	0.22	0.78	0.43	0.20	92.85	0.46	0.18	0.34	18.57
		0.50	0.34	0.34	0.06	0.40	92.48	0.17	0.03	0.41	18.46
multi	non-robust model	-	-	0.97	0.82	0.00	8.98	0.78	0.45	0.00	4.21
	base model	-	-	0.92	0.60	0.00	9.04	0.56	0.19	0.00	4.22
	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	0.88	0.57	0.09	1040.55	0.52	0.21	0.29	355.00

observe a sudden drop-off as n is becomes insufficient to proof any component (even those that are always correct). All values use Holm correction. We contrast them with Bonferroni correction in App. B.3.

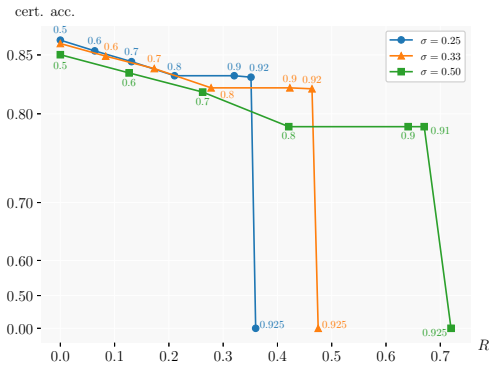


Figure 3. Radius versus certified mean per-pixel accuracy for semantic segmentation on Cityscapes at scale 0.25. Numbers next to dots show τ. The y-axis is scaled to the fourth power for clarity.

Lastly, we observe that our algorithm comes with a large increase in run-time as is common with randomized smoothing. However, the effect here is amplified as semantic segmentation commonly considers large models and large images. On average, 30s (for $n = 100$) are spent on computing the p-values, 0.1s on computing the Holm correction and the test of the reported time on sampling. We did not optimize our implementation for speed, other than choosing the largest possible batch size for each scale, and we believe that with further engineering run time can be reduced.

6.3. Pointcloud Part Segmentation

For point clouds part segmentation we use the ShapeNet (Chang et al., 2015) dataset, which contains $3d$ CAD models of different objects from 16 categories, with multiple parts, annotated from 50 labels. Given the overall label and points sampled from the surface of the object, the goal is to segment them into their correct parts. There exist two variations of the described ShapeNet task, one where every point is a $3d$ coordinate and one where each coordinate also

Table 2. results on 100 point clouds part segmentations. acc. shows the mean per-point accuracy, %⊘ abstentions and t runtime in seconds. all SEGCERTIFY ($n_0 = 100, \alpha = 0.001$) results are certifiably robust w.h.p.

	n	τ	σ	acc	%⊘	t
$f_{\sigma=0.25}$	-	-	-	0.81	0.00	0.57
	1000	0.75	0.250	0.62	0.32	54.47
	1000	0.85	0.250	0.52	0.44	54.41
	10000	0.95	0.250	0.41	0.56	496.57
	10000	0.99	0.250	0.20	0.78	496.88
$f^n_{\sigma=0.25}$	-	-	-	0.86	0.00	0.57
	1000	0.75	0.250	0.78	0.16	54.46
	1000	0.85	0.250	0.71	0.25	54.41
	10000	0.95	0.250	0.62	0.35	496.65
	10000	0.99	0.250	0.39	0.60	496.68
$f_{\sigma=0.5}$	-	-	-	0.70	0.00	0.55
	1000	0.75	0.250	0.57	0.23	54.40
	1000	0.75	0.500	0.47	0.44	54.47
	1000	0.85	0.500	0.41	0.54	54.39
	10000	0.95	0.500	0.30	0.67	496.30
	10000	0.99	0.500	0.11	0.87	496.47
$f^n_{\sigma=0.5}$	-	-	-	0.78	0.00	0.56
	1000	0.75	0.250	0.74	0.10	54.78
	1000	0.75	0.500	0.67	0.21	54.58
	1000	0.85	0.500	0.61	0.30	54.44
	10000	0.95	0.500	0.53	0.41	497.40
	10000	0.99	0.500	0.37	0.60	497.87

contains its surface normal vector. Each input consists of 2048 points and before it is fed to the neural network its coordinates are centered at the origin and scaled such that the point furthest away from the origin has distance 1.

Using the PointNetV2 architecture (Qi et al., 2017a;b; Yan et al., 2020), we trained four base models: $f_{\sigma=0.25}$, $f^n_{\sigma=0.25}$, $f^n_{\sigma=0.5}$ and $f_{\sigma=0.5}$ where $\cdot^n$ denotes the inclusion of normal vectors and σ the noise used in training. We apply all smoothing after the rescaling to work on consistent sizes and we only perturb the location, not the normal vector. Commonly, the underlying network is invoked 3 times and the results averaged as the classification algorithm involves randomness. As Theorems 3.1 and 5.1 do not require the underlying f to be deterministic, we also use the average of 3 runs for each invocation of the base model.

We provide the results in Table 2 and Fig. 4. We note that on the same data non-robust model achieves 0.91 and 0.90 with and without normal vectors respectively. While certification results here are good compared with the base model, this ratio is worse than in image segmentation and more samples (n) are needed. This is because here – in contrast to image segmentation – a perturbed point can have different semantics if it is moved to a different part of the object, causing label fluctuation. Further, as Fig. 4, shows we see a gradual decline in accuracy when increasing τ rather than the sudden drop in Fig. 3. As before, all values

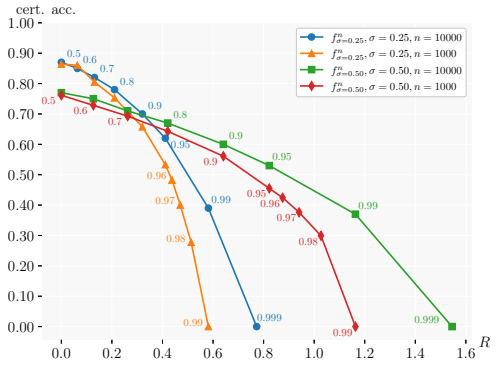


Figure 4. Radius versus certified accuracy at different radii for Pointcloud part segmentation. Numbers next to dots show τ.

are computed using Holm correction and we provide results for Bonferroni in App. B.4.

Using the same base models, we apply the randomized smoothing variant by Fischer et al. (2020) and achieve an accuracy of 0.69 while showing robustness to 3d rotations. App. B.5 provides further details and the obtained guarantee.

6.4. Discussion & Limitation

By reducing the problem of certified segmentation to only non-fluctuating components, we significantly reduce the difficulty and achieve strong results on challenging datasets. However, a drawback of the method is the newly introduced hyperparameter τ. In practice a suitable value can be determined by the desired radius or the empirical performance of the base classifier. High values of τ will permit a higher certification radius, but also lead to more abstentions and require more samples n for both certification and inference (both done via SEGCERTIFY). Further, as is common with adversarial defenses is a loss of performance compared to a non-robust model. However we expect further improvements in accuracy by the application of specialized training procedures (Salman et al., 2019; Zhai et al., 2020; Jeong & Shin, 2020), which we briefly investigate in App. B.3.

7. Conclusion

In this work we investigated the problem of provably robust segmentation algorithms and identified two key challenges: bad components and trade-offs introduced by multiple testing that prevent naive solutions from working well. Based on this we introduced SEGCERTIFY, a simple yet powerful algorithm that clearly overcomes the bad component issue and allows for significantly better trade-offs in multiple-testing. It enables certified performance on a similar level as an undefended base classifier and permits guarantees to multiple threat models in a plug-and-play manner.

References

Arnab, A., Miksik, O., and Torr, P. H. S. On the robustness of semantic segmentation models to adversarial attacks. In *CVPR*, 2018.

Balunovic, M. and Vechev, M. T. Adversarial training and provable defenses: Bridging the gap. In *ICLR*, 2020.

Balunovic, M., Baader, M., Singh, G., Gehr, T., and Vechev, M. T. Certifying geometric robustness of neural networks. In *NeurIPS*, 2019.

Bielik, P. and Vechev, M. T. Adversarial robustness for code. In *ICML*, 2020.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Srndic, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In *ECML/PKDD*, 2013.

Bojchevski, A., Klicpera, J., and Günnemann, S. Efficient robustness certificates for discrete data: Sparsity-aware randomized smoothing for graphs, images and more. In *ICML*, 2020.

Bonferroni, C. Teoria statistica delle classi e calcolo delle probabilita. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commericiali di Firenze*, 8:3–62, 1936.

Chang, A. X., Funkhouser, T. A., Guibas, L. J., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., and Yu, F. Shapenet: An information-rich 3d model repository. Technical report, 2015.

Chiang, P., Curry, M. J., Abdelkader, A., Kumar, A., Dickerson, J., and Goldstein, T. Detection as regression: Certified object detection with median smoothing. In *NeurIPS*, 2020.

Clopper, C. J. and Pearson, E. S. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26(4):404–413, 1934.

Cohen, J. M., Rosenfeld, E., and Kolter, J. Z. Certified adversarial robustness via randomized smoothing. In *ICML*, 2019.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

Deng, L., Yang, M., Qian, Y., Wang, C., and Wang, B. CNN based semantic segmentation for urban traffic scenes using fisheye camera. In *Intelligent Vehicles Symposium*. IEEE, 2017.

Ehlers, R. Formal verification of piece-wise linear feed-forward neural networks. In *ATVA*, 2017.

Fischer, M., Baader, M., and Vechev, M. T. Certified defense to image transformations via randomized smoothing. In *NeurIPS*, 2020.

Gehr, T., Mirman, M., Drachsler-Cohen, D., Tsankov, P., Chaudhuri, S., and Vechev, M. T. AI2: safety and robustness certification of neural networks with abstract interpretation. In *IEEE Symposium on Security and Privacy*, 2018.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T. A., and Kohli, P. On the effectiveness of interval bound propagation for training verifiably robust models. *CoRR*, abs/1810.12715, 2018.

Holm, S. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pp. 65–70, 1979.

Jeong, J. and Shin, J. Consistency regularization for certified robustness of smoothed classifiers. In *NeurIPS*, 2020.

Katz, G., Barrett, C. W., Dill, D. L., Julian, K., and Kochenderfer, M. J. Reluplex: An efficient SMT solver for verifying deep neural networks. In *CAV*, 2017.

Kurakin, A., Goodfellow, I. J., and Bengio, S. Adversarial machine learning at scale. In *ICLR*, 2017.

Lécuyer, M., Atlidakis, V., Geambasu, R., Hsu, D., and Jana, S. Certified robustness to adversarial examples with differential privacy. In *IEEE Symposium on Security and Privacy*, 2019.

Lehmann, E. L. and Romano, J. P. Generalizations of the familywise error rate. In *Selected Works of EL Lehmann*, pp. 719–735. Springer, 2012.

Levine, A. and Feizi, S. (de)randomized smoothing for certifiable defense against patch attacks. In *NeurIPS*, 2020a.

Levine, A. and Feizi, S. Wasserstein smoothing: Certified robustness against wasserstein adversarial attacks. In *AISTATS*, 2020b.

Li, B., Chen, C., Wang, W., and Carin, L. Second-order adversarial attack and certifiable robustness. *CoRR*, abs/1809.03113, 2018.

Li, L., Weber, M., Xu, X., Rimanic, L., Kailkhura, B., Xie, T., Zhang, C., and Li, B. Tss: Transformation-specific smoothing for robustness certification. In *CCS*, 2021.

Lin, W.-Y., Sheikholeslami, F., jinghao shi, Rice, L., and Kolter, J. Z. Certified robustness against physically-realizable patch attack via randomized cropping, 2021.

Liu, D., Yu, R., and Su, H. Extending adversarial attacks and defenses to deep 3d point cloud classifiers. In *ICIP*, 2019.

Lorenz, T., Ruoss, A., Balunovic, M., Singh, G., and Vechev, M. T. Robustness certification for point cloud models. *CoRR*, abs/2103.16652, 2021.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Mirman, M., Gehr, T., and Vechev, M. T. Differentiable abstract interpretation for provably robust neural networks. In *ICML*, 2018.

Mohapatra, J., Ko, C., Weng, T., Chen, P., Liu, S., and Daniel, L. Higher-order certification for randomized smoothing. In *NeurIPS*, 2020a.

Mohapatra, J., Weng, T., Chen, P., Liu, S., and Daniel, L. Towards verifying robustness of neural networks against A family of semantic perturbations. In *CVPR*, 2020b.

Mottaghi, R., Chen, X., Liu, X., Cho, N., Lee, S., Fidler, S., Urtasun, R., and Yuille, A. L. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

Perone, C. S., Calabrese, E., and Cohen-Adad, J. Spinal cord gray matter segmentation using deep dilated convolutions. *Scientific reports*, 8(1):1–13, 2018.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017a.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 2017b.

Raghunathan, A., Steinhardt, J., and Liang, P. Semidefinite relaxations for certifying robustness to adversarial examples. In *NeurIPS*, 2018a.

Raghunathan, A., Steinhardt, J., and Liang, P. Certified defenses against adversarial examples. In *ICLR*, 2018b.

Ruoss, A., Baader, M., Balunovic, M., and Vechev, M. T. Efficient certification of spatial robustness, 2021.

Salman, H., Li, J., Razenshteyn, I. P., Zhang, P., Zhang, H., Bubeck, S., and Yang, G. Provably robust deep learning via adversarially trained smoothed classifiers. In *NeurIPS*, 2019.

Salman, H., Sun, M., Yang, G., Kapoor, A., and Kolter, J. Z. Denoised smoothing: A provable defense for pretrained classifiers. In *NeurIPS*, 2020.

Savitzky, A. and Golay, M. J. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639, 1964.

Schuchardt, J., Bojchevski, A., Klicpera, J., and Günnemann, S. Collective robustness certificates. In *ICLR*, 2021.

Sheikholeslami, F., Lotfi, A., and Kolter, J. Z. Provably robust classification of adversarial examples with detection. In *ICLR*, 2021.

Šidák, Z. Rectangular confidence regions for the means of multivariate normal distributions. *Journal of the American Statistical Association*, 62(318):626–633, 1967.

Singh, G., Ganvir, R., Püschel, M., and Vechev, M. T. Beyond the single neuron convex barrier for neural network certification. In *NeurIPS*, 2019a.

Singh, G., Gehr, T., Püschel, M., and Vechev, M. T. An abstract domain for certifying neural networks. *Proc. ACM Program. Lang.*, (POPL), 2019b.

Sun, J., Cao, Y., Chen, Q. A., and Mao, Z. M. Towards robust lidar-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures. In *USENIX Security Symposium*, 2020.

Sun, K., Xiao, B., Liu, D., and Wang, J. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R. Intriguing properties of neural networks. In *ICLR*, 2014.

Tran, D., Pal, N., Musau, P., Manzanas Lopez, D., Hamilton, N., Yang, X., Bak, S., and Johnson, T. Robustness verification of semantic segmentation neural networks using relaxed reachability. In *CAV*, 2021.

Tukey, J. W. Comparing individual means in the analysis of variance. *Biometrics*, pp. 99–114, 1949.

Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., Liu, W., and Xiao, B. Deep high-resolution representation learning for visual recognition. *TPAMI*, 2019.

Wang, S., Pei, K., Whitehouse, J., Yang, J., and Jana, S. Efficient formal safety analysis of neural networks. In *NeurIPS*, 2018.

Weng, T., Zhang, H., Chen, H., Song, Z., Hsieh, C., Daniel, L., Boning, D. S., and Dhillon, I. S. Towards fast computation of certified robustness for relu networks. In *ICML*, 2018.

Westfall, P. Simultaneous small-sample multivariate bernoulli confidence intervals. *Biometrics*, pp. 1001–1013, 1985.

Westfall, P. H. and Young, S. S. *Resampling-based multiple testing: Examples and methods for p-value adjustment*, volume 279. John Wiley & Sons, 1993.

Wong, E. and Kolter, J. Z. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, 2018.

Xiang, C., Qi, C. R., and Li, B. Generating 3d adversarial point clouds. In *CVPR*, 2019.

Xie, C., Wang, J., Zhang, Z., Zhou, Y., Xie, L., and Yuille, A. L. Adversarial examples for semantic segmentation and object detection. In *ICCV*, 2017.

Yan, X., Zheng, C., Li, Z., Wang, S., and Cui, S. Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling. In *CVPR*, 2020.

Yang, G., Duan, T., Hu, J. E., Salman, H., Razenshteyn, I. P., and Li, J. Randomized smoothing of all shapes and sizes. In *ICML*, 2020.

Zhai, R., Dan, C., He, D., Zhang, H., Gong, B., Ravikumar, P., Hsieh, C., and Wang, L. MACER: attack-free and scalable robust training via maximizing certified radius. In *ICLR*, 2020.

Zhang, H., Weng, T., Chen, P., Hsieh, C., and Daniel, L. Efficient neural network robustness certification with general activation functions. In *NeurIPS*, 2018.

Supplemental Material for Scalable Certified Segmentation via Randomized Smoothing

A. Experimental Details

A.1. Experimental Details for §6.2

We use a HrNetV2 (Sun et al., 2019; Wang et al., 2019) with the HRNetV2-W48 backbone from their official PyTorch 1.1 (Paszke et al., 2019) implementation[1]. For Cityscapes we follow the outlined training procedure, only adding the $\sigma = 0.25$ Gaussian noise. For Pascal Context we doubled the number of training epochs (and learning rate schedule) and added the $\sigma = 0.25$ Gaussian noise. During inference we use different batch sizes for different scales. These are summarized in Table 3. All timing timing results are given for a single Nvidia GeForce RTX 2080 Ti and using 12 cores of a Intel(R) Xeon(R) Silver 4214R CPU @ 2.40GHz. When training on an machine with 8 Nvidia GeForce RTX 2080 Ti one training epoch takes around 4 minutes for both of the data sets.

We evaluate on 100 images each, that is for Cityscapes we use every 5th image in the test set and for Pascal every 51st.

We consider two metrics:

- (certified) per-pixel accuracy: the rate of pixels correctly classified (over all images)
- (certified) mean intersection over union (mIoU): For each image i and each class c (ignoring the $\oslash$ "class") we calculate the ratio $IoU_c^i = \frac{|P_c^i \cap G_c^i|}{|P_c^i \cup G_c^i|}$ where P_c^i denotes the pixel locations predicted as class c for input i and G_c^i denotes the pixel locations for class c in the ground truth of input i. We then average IoU_c^i over all inputs and classes.

A.2. Experimental Details for §6.3

Using the PointNetV2 architecture (Qi et al., 2017a;b; Yan et al., 2020) implemented in PyTorch [2] (Paszke et al., 2019). Again we keep the training parameters unchanged other than the addition of noise during training. One training epoch on a single Nvidia GeForce RTX 2080 Ti takes 6 minutes.

[1] https://github.com/HRNet/HRNet-Semantic-Segmentation

[2] https://github.com/yanx27/Pointnet_Pointnet2_pytorch

Table 3. Batch sizes used in segmentation inference.

scale	Cityscapes	Pascal Context
0.25	24	80
0.50	12	64
0.75	4	32
1.00	4	20
multi	4	10

In inference we use a batch size of 50. All timing results are given for a single Nvidia GeForce RTX 2080 Ti and using 12 cores of a Intel(R) Xeon(R) Silver 4214R CPU @ 2.40GHz.

Again, we evaluate on 100 inputs. This corresponds to every 28th input in the test set. As a metric we consider the (certified) per-component accuracy: the rate of parts/components correctly classified (over all inputs).

B. Additional Results

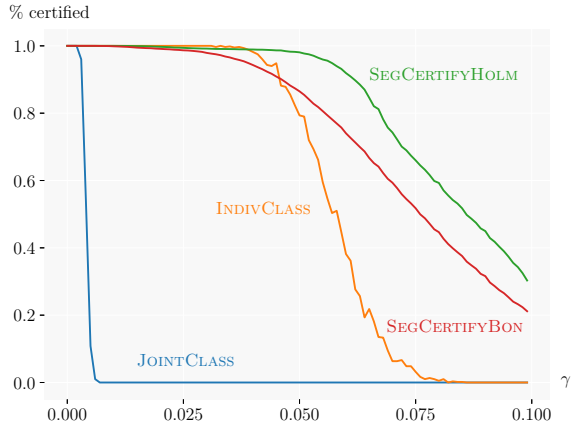
B.1. Additional Results for §6.1

In Figs. 2b and 2c we show smoothed plots as JOINTCLASS and INDIVCLASS are either correct on all components or non at all. Here, we provide the unsmoothed results in Fig. 5. In order to obtain the plots in Fig. 2 we apply a Savgol filter (Savitzky & Golay, 1964) of degree 1 over the 11 closest neighbours (using the SciPy implementation) and use a step size of 0.001 for γ.

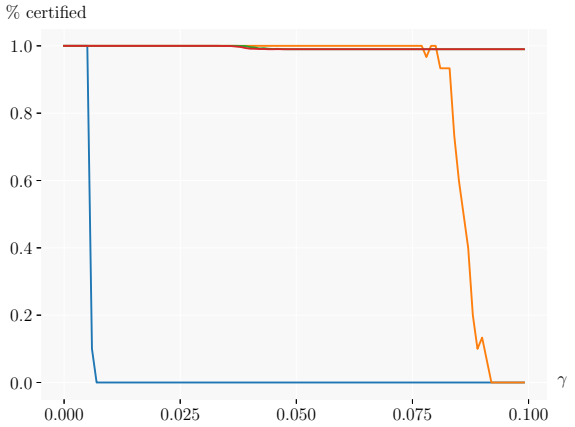
B.2. k-FWER and error budget

Here we discuss the gains from allowing a small budget of errors and applying k-FWER control as outlined in §5.2. Control for k-FWER at level α means that $P(\geq k \text{ type I errors}) \leq \alpha$. Which for $k = 1$ recovers standard FWER control. Thus, if we allow a budget of b type I errors at level α we need to perform k-FWER control with $k = b + 1$. In the following we will refer only to the budget b, to avoid confusion between the k in k-FWER and the k noisy components in the setting of §6.1.

Fig. 6 shows an empirical evaluation of this approach for different b for $\gamma = 0.05$, one noisy components and different

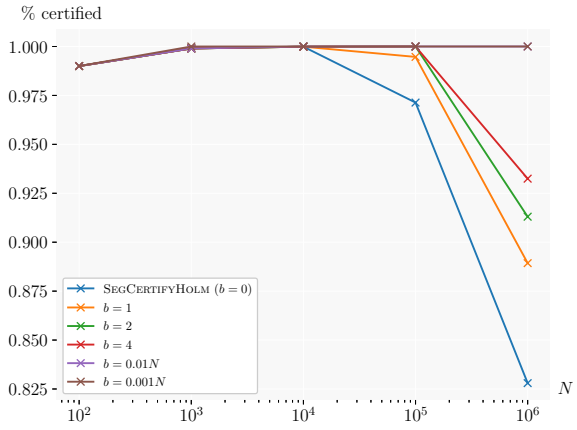


(a) Unsmoothed version of Fig. 2b. On $N = 100$ components, with a classifier that has error rate 5γ on one component and γ on all others.

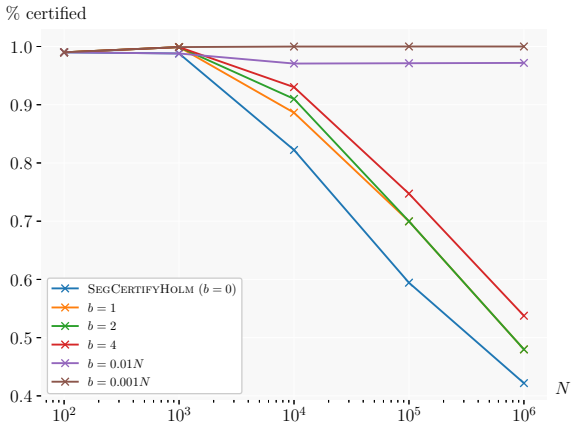


(b) Unsmoothed version of Fig. 2c. SegCertify with different testing corrections for various numbers of components N with error rate $\gamma = 0.05$.

Figure 5. Unsmoothed versions of Fig. 2. We empirically investigate the power (ability to avoid type II errors – false abstention) of multiple algorithms on synthetic data. The y-axis shows the rate of certified (rather than abstained) components. An optimal algorithm would achieve 1.0 or 0.99 in all plots.



(a) $\alpha = 0.1$.



(b) $\alpha = 0.001$.

Figure 6. Evaluation where an error budget of up top b type I errors is allowed. Potentially allowing a small amount of errors greatly increases the power of the test. N varies along the x-axis, $\gamma = 0.05$ and k components with error rate 5γ.

levels of N. Fig. 6a uses $\alpha = 0.1$ and Fig. 6b $\alpha = 0.001$. We see that allowing a single error leads to a huge gain in power for the $N = 10^6$ setting. Similarly, allowing 1 percent or 1 permille errors greatly strengthen the method.

False Discovery Rate Similarly, SegCertify can employ false discovery rate (FDR) control rather than (k-)FWER control. FDR control limits the expected number of type I errors. Since this is a much weaker statement than FWER it allows for less type II errors. However, while useful this kind of control leaves the area of (statistical) certified robustness for a more relaxed probabilistic setting which we do not investigate here further.

B.3. Additional Results for §6.2

Table 8 shows an extended version of Table 1. Both of these tables use Holm correction. Table 4 shows the difference to Table 8 if instead Bonferroni correction was used. Generally this difference is very small in this setting as it appears the true p_A are far from τ. However in some settings (such as multi resolution) the effect of Holm correction can be up to 2%. Since for a particular base classifier or τ this gain can quickly go from neglectable to significant and since the additional evaluation time ($< 0.1s$) is neglectable compared to the time for sampling we believe Holm correction to be preferable in most cases.

Table 4. Difference when Bonferroni correction rather than Holm correction is used in Table 8. Only differences $\geq 10^{-4}$ are shown. We observed no such differences on the Pascal Context dataset.

				Cityscapes		
scale		σ	R	acc.	mIoU	%⊘
0.5	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	-0.0008	-0.0005	0.0019
		0.33	0.22	-0.0014	-0.0010	0.0024
		0.50	0.34	-0.0021	-0.0019	0.0031
0.5	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	-0.0009	-0.0014	0.0014
		0.33	0.22	-0.0012	-0.0020	0.0016
		0.50	0.34	-0.0005	-0.0004	0.0006
1.0	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	-0.0012	-0.0032	0.0016
		0.33	0.22	-0.0020	-0.0018	0.0024
		0.50	0.34	-0.0000	-0.0000	0.0001
	SEGCERTIFY $n = 300, \tau = 0.90$	0.25	0.17	0.0001	0.0002	-0.0002
multi	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	-0.0125	-0.0185	0.0173

Consistency Training Here we investigate a naive instantiation of training approach from Jeong & Shin (2020).

Jeong & Shin (2020) improve the training for classification models used as base models in randomized smoothing by adding a consistency regularization term in training. We use this same term, but compute it for every pixel and average the results. To obtain the results in Table 5 we used $m = 2$, $\lambda = 1$, $\eta = 0.5$ and $\sigma = 0.25$. Depending on the scale we see either slightly better or slightly worse results than with standard Gaussian data augmentation in training. This shows the promise of the method but also highlights the need for potential further specialization to the segmentation setting, particularly by considering the effect of scaling.

B.4. Additional Results for §6.3

Table 6 shows the change when executing the experiments in Table 2 with Bonferroni correction instead of Holm correction.

B.5. Certification beyond ℓ_p

As outlined in §5.2, SEGCERTIFY can be easily adapted to non-ℓ_2-settings. Here we show that we can certify against and adversary rotating 3d point clouds. A 3d rotation is parameterized by 3 angles which we will denote $\epsilon \in \mathbb{R}^3$ and define $\psi_\epsilon(\boldsymbol{x})\colon \mathbb{R}^{N\times 3} \to \mathbb{R}^{N\times 3}$ as

$$(\psi_\epsilon(\boldsymbol{x}))_i = \boldsymbol{R}_\epsilon \boldsymbol{x}_i, \tag{2}$$

where $\boldsymbol{R}_\epsilon$ denotes the 3d rotation matrix specified by ϵ. The randomized smoothing approach of Fischer et al. (2020) allows to certify robustness in this case: $f(\boldsymbol{x}) = f(\psi_\delta(\boldsymbol{x}))$ for δ with $\|\delta\|_2 \leq R$, by sampling rotations $\psi_\epsilon(\boldsymbol{x})$ with $\epsilon \sim \mathcal{N}(0, \sigma^2)$. The parameter robustness radius R is computed the same as throughout the paper. When applied to points with a normal vector, Eq. (2) can be extended to apply R_ϵ to the point coordinates as well as the normal vector.

Using one of the model from §6.3, $f^n_{\sigma=0.5}$, we perform this version of randomized smoothing.

The results are shown in Table 7. Since the models was not specifically trained to be robust under rotations, the performance quickly deteriorates. Nevertheless we can certify robustness to rotations with a parameter radius R of 0.17 and 0.085 for σ of 0.25 and 0.125 respectively.

The same approach can be applied to models that are empirically invariant to most rotations while not formally rotation invariant. In these cases we need to certify a radius of $R = \sqrt{3}\pi$ (when measuring angles in radians). When using a fixed τ, an appropriate σ can be chosen as $\sigma = \frac{\sqrt{3}\pi}{\Phi^{-1}(\tau)}$. While this is relativity large σ, this does not pose an obstacle for a mostly robust base model.

Table 7. Results for point cloud part segmentation under 3d rotation. The baseline and base model is $f^n_{\sigma=0.5}$. SEGCERTIFY uses $\tau = 0.75$, $n_0 = 100$, $n = 1000$ and $\alpha = 0.0001$.

model / σ	acc	%⊘	t
baseline	0.77	0.00	0.72
0.125	0.69	0.16	74.13
0.25	0.61	0.26	74.51

Table 5. Same setting as Table 8 but using a model trained with consitency regularization.

				Cityscapes		
scale		σ	R	acc.	mIoU	%⊘
0.25	base model	-	-	0.87	0.40	0.00
	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	0.83	0.42	0.07
		0.33	0.22	0.84	0.42	0.09
		0.50	0.34	0.82	0.43	0.14
0.50	base model	-	-	0.91	0.53	0.00
	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	0.89	0.57	0.06
		0.33	0.22	0.89	0.57	0.07
		0.50	0.34	0.86	0.57	0.11
1.00	base model	-	-	0.92	0.62	0.00
	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	0.88	0.63	0.12
		0.33	0.22	0.79	0.46	0.20
		0.50	0.34	0.39	0.06	0.32

Table 6. Difference when Bonferroni correction rather than Holm correction is used in Table 2.

	n	τ	σ	acc	%⊘
$f_{\sigma=0.25}$					
	1000	0.75	0.250	-0.0012	0.0019
	1000	0.85	0.250	-0.0023	0.0029
	10000	0.95	0.250	-0.0013	0.0009
	10000	0.99	0.250	-0.0008	0.0009
$f^n_{\sigma=0.25}$					
	1000	0.75	0.250	-0.0020	0.0026
	1000	0.85	0.250	-0.0026	0.0032
	10000	0.95	0.250	-0.0011	0.0010
	10000	0.99	0.250	-0.0013	0.0011
$f_{\sigma=0.5}$					
	1000	0.75	0.250	-0.0007	0.0010
	1000	0.75	0.500	-0.0002	0.0010
	1000	0.85	0.500	-0.0017	0.0028
	10000	0.95	0.500	-0.0007	0.0010
	10000	0.99	0.500	-0.0003	0.0003
$f^n_{\sigma=0.5}$					
	1000	0.75	0.250	-0.0009	0.0017
	1000	0.75	0.500	-0.0015	0.0024
	1000	0.85	0.500	-0.0019	0.0028
	10000	0.95	0.500	-0.0043	0.0013
	10000	0.99	0.500	-0.0008	0.0007

Table 8. Extended version of Table 1. Segmentation results for 100 images. acc. shows the mean per-pixel accuracy, mIoU the mean intersection over union, %⊘ abstentions and t runtime in seconds. All SEGCERTIFY ($n_0 = 10, \alpha = 0.001$) results are certifiably robust at radius R w.h.p. multiscale uses $0.5, 0.75, 1.0, 1.25, 1.5, 1.75$ as well as their flipped variants for Cityscapes and additionally 2.0 for Pascal. All numbers are obtained via Holm correction.

				Cityscapes				Pascal Context			
scale		σ	R	acc.	mIoU	%⊘	t	acc.	mIoU	%⊘	t
0.25	non-robust model	-	-	0.93	0.60	0.00	0.38	0.59	0.24	0.00	0.12
	base model	-	-	0.87	0.42	0.00	0.37	0.33	0.08	0.00	0.13
	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	0.84	0.43	0.07	70.00	0.33	0.08	0.13	14.16
		0.33	0.22	0.84	0.44	0.09	70.21	0.34	0.09	0.17	14.20
		0.50	0.34	0.82	0.43	0.13	71.45	0.23	0.05	0.27	14.23
	SEGCERTIFY $n = 300, \tau = 0.90$	0.25	0.32	0.83	0.43	0.10	143.37	0.32	0.08	0.23	24.33
		0.33	0.42	0.82	0.43	0.12	143.30	0.32	0.09	0.29	24.42
		0.50	0.64	0.79	0.40	0.18	143.54	0.20	0.04	0.43	24.13
	SEGCERTIFY $n = 500, \tau = 0.95$	0.25	0.41	0.83	0.42	0.11	229.37	0.29	0.01	0.30	33.64
		0.33	0.52	0.83	0.42	0.12	230.69	0.26	0.01	0.39	33.79
		0.50	0.82	0.77	0.38	0.20	230.09	0.10	0.00	0.61	33.44
	SEGCERTIFY $n = 10000, \tau = 0.99$	0.25	0.58	-	-	-	-	0.25	0.07	0.48	557.29
		0.33	0.77	-	-	-	-	0.24	0.07	0.58	557.34
		0.50	1.17	-	-	-	-	0.11	0.03	0.77	557.32
0.5	non-robust model	-	-	0.96	0.76	0.00	0.39	0.74	0.38	0.00	0.16
	base model	-	-	0.89	0.51	0.00	0.39	0.47	0.13	0.00	0.14
	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	0.88	0.54	0.06	75.59	0.48	0.16	0.09	16.29
		0.33	0.22	0.87	0.54	0.08	75.99	0.50	0.17	0.11	16.08
		0.50	0.34	0.86	0.54	0.10	75.72	0.36	0.10	0.21	16.14
	SEGCERTIFY $n = 300, \tau = 0.90$	0.25	0.32	0.87	0.53	0.08	143.40	0.46	0.15	0.17	27.17
		0.33	0.42	0.86	0.52	0.10	145.90	0.47	0.16	0.21	27.17
		0.50	0.64	0.83	0.50	0.15	144.61	0.31	0.10	0.38	27.32
	SEGCERTIFY $n = 500, \tau = 0.95$	0.25	0.41	0.86	0.52	0.09	228.63	0.45	0.19	0.21	38.27
		0.33	0.52	0.85	0.51	0.11	228.38	0.46	0.16	0.26	38.43
		0.50	0.82	0.82	0.49	0.16	228.73	0.30	0.09	0.44	38.37
0.75	non-robust model	-	-	0.97	0.80	0.00	0.46	0.76	0.41	0.00	0.15
	base model	-	-	0.90	0.59	0.00	0.47	0.55	0.18	0.00	0.15
	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	0.88	0.57	0.09	82.69	0.53	0.19	0.15	16.78
		0.33	0.22	0.86	0.56	0.12	82.87	0.54	0.20	0.20	16.83
		0.50	0.34	0.64	0.27	0.31	82.19	0.29	0.07	0.33	16.83
	SEGCERTIFY $n = 300, \tau = 0.90$	0.25	0.32	0.84	0.54	0.13	177.44	0.51	0.19	0.22	29.48
		0.33	0.42	0.84	0.52	0.15	177.22	0.51	0.20	0.28	29.58
		0.50	0.64	0.60	0.24	0.37	177.67	0.25	0.06	0.45	29.53
1.0	non-robust model	-	-	0.97	0.81	0.00	0.52	0.77	0.42	0.00	0.18
	base model	-	-	0.91	0.57	0.00	0.52	0.53	0.18	0.00	0.18
	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	0.88	0.59	0.11	92.75	0.55	0.22	0.22	18.53
		0.33	0.22	0.78	0.43	0.20	92.85	0.46	0.18	0.34	18.57
		0.50	0.34	0.34	0.06	0.40	92.48	0.17	0.03	0.41	18.46
	SEGCERTIFY $n = 300, \tau = 0.90$	0.25	0.32	0.86	0.56	0.14	204.82	0.53	0.21	0.29	33.83
		0.33	0.42	0.75	0.40	0.24	204.58	0.42	0.17	0.44	33.78
		0.50	0.64	0.31	0.05	0.47	204.57	0.15	0.03	0.52	33.43
multi	non-robust model	-	-	0.97	0.82	0.00	8.98	0.78	0.45	0.00	4.21
	base model	-	-	0.92	0.60	0.00	9.04	0.56	0.19	0.00	4.22
	SEGCERTIFY $n = 100, \tau = 0.75$	0.25	0.17	0.88	0.57	0.09	1040.55	0.52	0.21	0.29	355.00

C. Details on Attacks & Fig. 1

To produce the visualization in Fig. 1 we used a custom PGD attack described below and produced an ℓ_2 adversarial example within a range of 0.16. We performed certification with $n = 100, \alpha = 0.001, \sigma = 0.25$ and $\tau = 0.75$, which certifies robustness at an radius of $R = \sigma\Phi^{-1}(\tau) = 0.1686$. We perform this certification on the clean input and thus show robustness to the attack. As the non-robust model for Fig. 1c we used a pretrained HrNet from the same repository as outlined in App. A.1.

We use $k = 100$ steps for the attack and step size $s = \frac{10*R}{k}$. While scaling can be simply incorporated into the attack, we use scale 1.0 for both the attacked an the certified classifier.

We used an Nvidia Titan RTX to perform these attacks as the memory requirement exceeds that of a single Nvidia GeForce RTX 2080 Ti.

Here, in Figs. 7 and 8 we provide further visualizations like Fig. 1. The visualization in Fig. 7 is hand-picked (like Fig. 1), while the ones in Fig. 8 are chosen randomly from the evaluated images.

C.1. PGD for Segmentation

Following the work of Madry et al. (2018), Arnab et al. (2018) and Xie et al. (2017) we use a slightly generalized form of the untargeted ℓ_2 projected gradient decent (PGD) attack. This version is the same as untargeted PGD (Madry et al., 2018) in the classification setting, but we adapt the loss to be the average of all pixel losses as in Xie et al. (2017). Formally, for an input $\boldsymbol{x} \in \mathcal{X}^N = [0,1]^{N\times 3}$ with ground truth segmentation $\boldsymbol{y} \in \mathbb{Y}^N$ and model f a radius R and stepsize $s \in \mathbb{R}$ we produce an adversarial example $\boldsymbol{x}'_k$ after k steps.

$$\begin{aligned}\boldsymbol{x}'_0 &= \boldsymbol{x} + \epsilon, \qquad \epsilon \sim [0,1] \text{ with } \|\epsilon\|_2 \leq R \\ \boldsymbol{x}'_{i+1} &= c_{R,\boldsymbol{x}}(\boldsymbol{x}'_i + s\nabla_{\boldsymbol{x}'_i}\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}))\end{aligned}$$

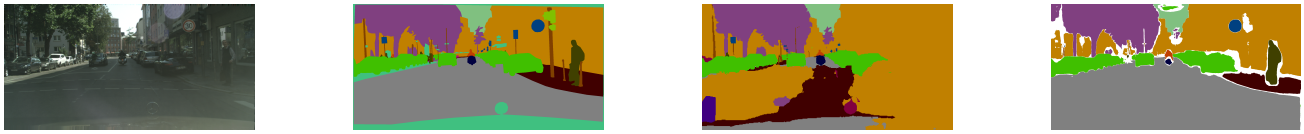
with clamping function

$$\begin{aligned}c_R &\colon \mathbb{R}^{N\times 3} \to [0,1]^{N\times 3} \\ c_{R,\boldsymbol{x}}(\boldsymbol{x}') &= \left[\boldsymbol{x} + R\frac{\boldsymbol{x}' - \boldsymbol{x}}{\|\boldsymbol{x}' - \boldsymbol{x}\|_2}\right],\end{aligned}$$

where $[\cdot]$ denotes component-wise clamping to $[0,1]$, and loss

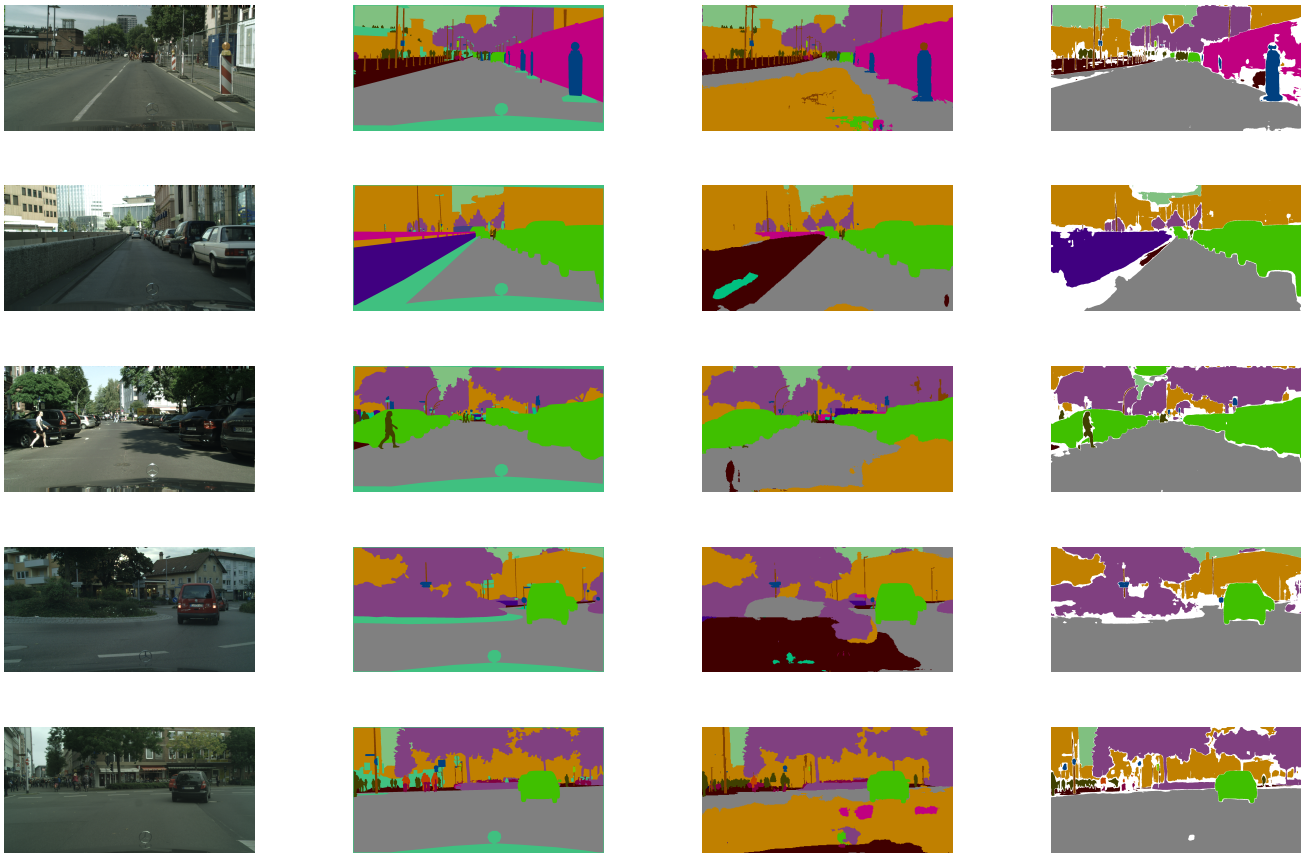
$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{H}(f_i(\boldsymbol{x}), \boldsymbol{y}_i), \tag{3}$$

where $\mathcal{H}$ denotes the cross entropy function.



(a) Attacked image (b) Ground truth seg. (c) Attacked segmentation (d) Certified segmentation

Figure 7. Another hand-picked example like Fig. 1.



(a) Attacked image (b) Ground truth seg. (c) Attacked segmentation (d) Certified segmentation

Figure 8. Randomly chosen examples like Fig. 1.